Automatic Mesh Coarsening for Discrete Ordinates Codes

Scott A. Turner
Los Alamos National Laboratory
MS D409, Los Alamos, NM 87545 (USA)
turners@lanl.gov

Abstract

This paper describes the use of a "mesh potential" function for automatic coarsening of meshes in discrete ordinates neutral particle transport codes. For many transport calculations, a user may find it helpful to have the code determine a "good" neutronics mesh. The complexity of a problem involving millions of mesh cells, dozens of materials, and many energy groups makes it difficult to determine an adequate level of mesh refinement with a minimum number of cells. A method has been implemented in PARTISN (PArallel TIme-dependent SN) to calculate a "mesh potential" in each original cell of a problem, and use this information to determine the maximum coarseness allowed in the mesh while maintaining accuracy in the solution. Results are presented for a simple x-y-z fuel/control/reflector problem.

1 Introduction

This paper describes a new parallel method for automatically determining a coarsened mesh that is sufficient for solving a neutral particle transport problem. Cross sections and original mesh sizes are used to determine a special type of mean free path (a mesh potential) that is used to decide which mesh planes may be removed while maintaining accuracy in the final solution. This method has been implemented in PARTISN (Alcouffe et al., 1998). PARTISN solves the linear Boltzmann transport equation for neutral particles using the deterministic (S_N) method on orthogonal (single level or block-structured AMR) grids in 1-D, 2-D (x-y, r-z, or r-t), and 3-D (x-y-z, r-z-t) geometries. PARTISN solves either the static or time-dependent forms of the transport equation, is accelerated with DSA, and is parallelized through the use of message passing (MPI). The automatic mesh coarsening method has been tested in all of the above modes.

The goal of this method is to allow a user to specify a simple uniform initial mesh, one that contains cells of equal width along each direction. The code automatically specifies a reduced mesh which may be nonuniform and more complicated than the original. Several candidates for the final mesh can be quickly generated. The user then has the option of examining each mesh before a solution is carried out, or of allowing the code to solve on each mesh as it is generated in order to observe the effects of mesh adjustment on the desired result. In the case of execution on a parallel architecture, load balancing occurs before and after the adjustment of the mesh, resulting in optimal parallel efficiency for both the analysis of the mesh and the solution on the reduced mesh.

Currently, the mesh potential is only utilized to automatically analyze and adjust simply connected meshes where each interior cell has exactly 6 neighbors. Simply connected meshes are inefficient because they do not allow very fine meshes in areas where large flux or source gradients are expected and very coarse meshes in areas where the flux and sources are expected to be flat. Refining any part of a simply connected mesh requires adding mesh planes that extend throughout the problem domain. Adaptive mesh refinement (AMR) involves meshes that are not simply connected and thus allow for potential improvements in the efficiency of the solution. In an AMR mesh, cells may have more than 6 neighbors, allowing for refinement in some areas of the mesh without affecting other areas. Such meshes are much more difficult to specify compared to a simply connected mesh. Block AMR, described later in this paper, is a simplified subset of arbitrary or cell-based AMR, but such meshes

are still difficult for a user to specify in an input deck. We have used the mesh adjustment routines of the mesh potential method to alleviate this burden on the user. As a result, block AMR meshes can be specified by the combination of a simple uniform mesh along with the desired level for each block. I will present the specification of a block AMR mesh, along with a graphical representation. The details of mesh potential and its implementation are covered in section 2. Strategies for utilization of the method and an example of its application are the subject of section 3. A discussion of the method's current level of usefulness, its limitations, and ideas for improvement are contained in section 4.

2 Implementation

The mesh potential function used in the automatic mesh adjustment routine has the following form:

$$P(i,j,k) = \left\{ \sum_{g=1}^{G} [w_g(a\sigma_{t,g} + b(\nu\sigma_f)_g)] \right\} max[\Delta x(i)\ ,\ \Delta y(j),\ \Delta z(k)] \qquad (1)$$

Here, w_g, a, and b, are user-defined weighting parameters. The sum of w_g over all energies is unity, the sum of a and b is unity, $\sigma_{t,g}$ and $\sigma_{f,g}$ are the total and fission cross sections for group g, and ν_g is the average number of neutrons released per fission. Coordinates for cells in the mesh are denoted by (i,j,k). A "large" value of $P(i,j,k)$, indicates that a cell may be too coarse for reasonable accuracy. A "small" value indicates that a cell may be eligible for coarsening in order to cut calculation time while maintaining reasonable accuracy. The user-defined parameters allow for the selection of which characteristic is to be emphasized. High values of a relative to b emphasize the total cross section, whereas high values of b relative to a emphasize the production of neutrons via fission. The distribution of w_g can be used to emphasize certain energy ranges. In addition, the user may also enter a lower bound for $P(i,j,k)$, below which coarsening of the mesh will occur.

Currently, two methods are used to estimate a good value for the minimum potential. The first method, and code's default, is to divide 0.5 by the number of energy groups. This default comes from the similarity between Eq. (1) and the traditional equation for the number of transport mean free paths, and the rule-of-thumb that one needs a mesh that is refined to about 0.5 mean free paths for diamond differencing. Defaults for other differencing schemes are being investigated. Division by the number of energy groups is suggested to account for the summation over all groups. The second method is trial and error. The users start with method 1, the default if a minimum potential is not entered, and adjust the result until they are satisfied with the accuracy of the solution. This method is only useful for determining a good value for similar problems that one intends to run, or if a user wants to investigate the affect of some parametric changes with minimal run time (i.e., determine the parameter that results in the coarsest "accurate" mesh; then use that for several subsequent runs in which small changes are made to various cross sections or configurations).

For 12 energy groups, a U235 fission spectrum for w_g, a = 0.1, and b = 0.9, the braced term in Eq. (1) gives a value of 0.173 for U235, 0.069 for U238, and 0.039 for water. With a = 0.9 and b = 0.1, the values are 0.290, 0.287, and 0.351 respectively. From these results, we can see that obtaining finer meshes in fissile materials and coarser meshes in nonfissile materials requires increased emphasis on the production of neutrons (b>0.5) and decreased emphasis on the total cross section (a<0.5).

The code analyzes the mesh and does one of three things, depending on an input control parameter. The first case merely calculates the mesh potential information for inclusion in the standard output file. The second case generates a file defining the suggested transformation from the initial mesh to a coarsened mesh. The third case analyzes the mesh, makes any adjustments suggested by the results, and continues to carry out the solution on the new mesh.

For simply connected meshes, the code calculates a potential for each cell of the mesh, and a maximum value is tallied for each plane of cells in each direction (x,y,z). A sweep is made along each axis, and the maximum value of each plane of cells on both sides of a mesh plane is tested against the user specified minimum. If the maximum value is less than the specified minimum, then the mesh plane is removed, the new width is used to calculate a new maximum value, and the test is conducted again. It is necessary to remove entire mesh planes in order to keep the mesh simply connected. The process of removing mesh planes continues until the maximum potential for a new plane of cells exceeds the specified minimum. Remixing of materials and recalculation of potentials does not occur during the coarsening sweep. This approach is conservative because remixing on the fly could result in more coarsening than without remixing. The method has been found capable of adequate coarsening without this added complication. For block AMR meshes (described in section 4), each AMR block of cells can be coarsened independently based on the user-defined level for each block.

In cases where entire planes of void exist, the user can choose, via an input parameter, whether or not the void planes should be collapsed into a single void or be left alone. Some users desire that external void planes be left alone for purposes of observing the flux at various points inside the void. In other cases, internal voids may require certain levels of refinement for accuracy of the solution.

The maximum aspect ratio (the ratio of the maximum and minimum cell dimensions) may also impact the solution. If the aspect ratio is too high in any region of the problem, then accuracy and efficiency may suffer. A solution to this problem would be to reinsert mesh planes in order to reduce the aspect ratio in those regions. Although this solution has not yet been implemented, information on the initial and final maximum aspect ratios is printed out.

The code has been tested on time-dependent time-of-flight problems where it checks the mesh at each timestep for adequacy. However, the mesh coarsening capability is not fully utilized in such problems because material properties do not change with time. Problems in which temperature changes result in time-dependent cross sections may benefit from time-dependent mesh adjustments. This case has been accounted for in the coding but has not been tested.

Adjustments to the mesh are achieved via the reallocation and mapping of several PARTISN arrays onto the new mesh in a conservative manner while maintaining load balance on a parallel architecture.

3 Utilization

In order to utilize the automatic coarsening capability, a user would first set up a problem with all the necessary details except the number of cells to be used in each zone. Next the user would determine a maximum number of cells for the entire problem based on memory and cpu time limitations. The resulting number of cells would then be uniformly assigned to each zone (i.e., a constant number of cells per cm). The problem is then run with user input controls set to only dump a file with specifications for coarsening the mesh. If the coarsening that resulted from use of default values does not appear acceptable, several more runs with various values for a and b in Eq. (1) and for various lower bounds on the mesh potential can be quickly executed (recall that only coarsening specifications are dumped to a file, no solution is being carried out).

How does one recognize an acceptable mesh? One purpose of this method is to allow dozens or even hundreds of similar problems to be solved quickly for the purpose of iterating to achieve some desired result (e.g., a specific dose limit, leakage, or k-effective). Thus, the user may start with a mesh that is known to be overly refined, obtain the solution on that mesh, then utilize the automatic coarsening capability to reduce the mesh to one that is solved much more quickly. With this in mind, an acceptable mesh is one that is a significant reduction of the original but that still results in a solution that is within a desired range of the original.

A second purpose for this method is to obtain a solution for a predetermined problem specification. In this case, materials, dimensions, and physical configurations are rigidly specified, but the number of mesh cells required for each zone is unknown. An acceptable mesh in this case is one in which the user has confidence. One way to determine confidence is to output the value of the mesh potential in each cell for subsequent graphical inspection. Several meshes could be quickly generated and examined graphically for regions with unacceptably high mesh potential values indicating an excessive level of coarseness.

For design purposes, once an acceptable mesh has been obtained, the same parameters can be used to obtain a solution by changing a user input control that allows the code to continue beyond coarsening with an actual solve on the reduced mesh. Various changes can then be made to the problem materials, geometry, etc., without concern for whether or not the mesh will be properly coarsened. Examples of changes that can be safely made include changes in fuel enrichment, changes in sizes and locations of fuel or control rods, changes in the thickness of reflectors or moderators, and others. Changes like these do not have a significant effect on how the mesh potential coarsens the mesh when using the same parameters that worked well on the original problem specification. This method is not proposed as a replacement for careful convergence studies (increasing quadrature order and decreasing mesh intervals until the answers no longer change significantly) to obtain accountable results. It is intended as a time-saving tool to be used during initial problem setup and subsequent design iterations.

The above procedure has been applied to the problem depicted in Figure 1 (Appendix A contains the actual PARTISN input deck). Characteristics of the problem include 2 energy groups, 100x100x100 uniformly distributed cells, 3 reflective boundaries, 3 vacuum boundaries, and cross sections which are modified versions of those used in OECD/NEA 3-D Neutron Transport Benchmark Model 1 (Takeda, 1991). For the purpose of demonstration, the fission cross section of the fuel, and the total cross section of the control rod have been exaggerated. No modifications were made to the polyethylene reflector. The mesh is divided into 5 zones in X and Y, and 3 zones in Z. Each zone dimension is 5 cm, except the middle zone in Z which is 15 cm long.

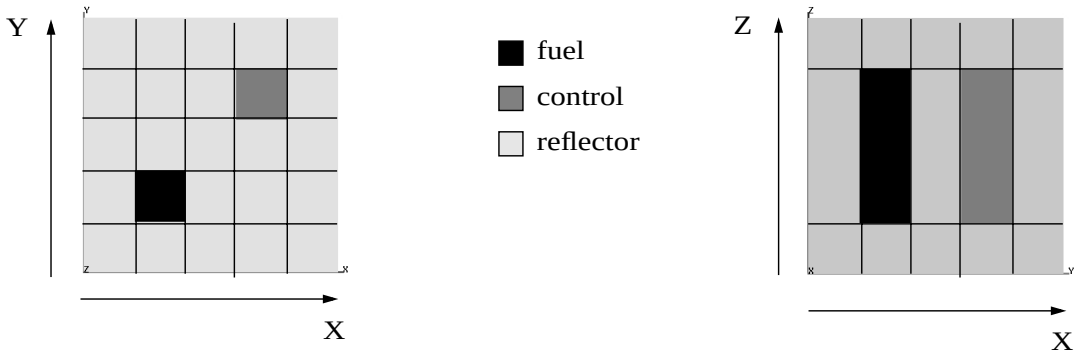

Figure 1. Top and Side Views of Material Zones.

An S_8 k-eigenvalue calculation using DSA with conjugant gradient preconditioning produced the results in Table 1 using 4 processors of an SGI Origin 2000. 20*1 indicates that 20 cells of the original mesh were left intact, where as “6*3 2” indicates six new cells containing 3 old cells followed by a new cell with 2 old cells (i.e., 7 new cells make up a zone that originally contained 20 old cells). Figures 2 through 5 display the affect of coarsening for 4 of the problems in Table 1. The original mesh, as noted above, contained a uniform distribution of an equal number of cells in each direction.

Table 1: Mesh Potential Parameters and Results

problem	a	b	lower bound and energy weights	X mesh cell relation by zone (total)	Y mesh cell relation by zone (total)	Z mesh cell relation by zone (total)	cpu iteration time in seconds	k-eff eigenvalue
1	n/a	n/a	n/a	20*1 20*1 20*1 20*1 20*1 (100)	20*1 20*1 20*1 20*1 20*1 (100)	20*1 60*1 20*1 (100)	1402	0.38208
2	0.9	0.1	0.25 and 0.5 0.5	10*2 10*2 10*2 20*1 10*2 (60)	10*2 10*2 10*2 20*1 10*2 (60)	10*2 60*1 10*2 (80)	237	0.38197
3	0.9	0.1	0.75 and 0.5 0.5	5*4 5*4 5*4 10*2 5*4 (30)	5*4 5*4 5*4 10*2 5*4 (30)	5*4 30*2 5*4 (40)	18	0.38156
4	0.2	0.8	0.25 and 0.5 0.5	3*6 2 6*3 2 3*6 2 10*2 3*6 2 (29)	3*6 2 6*3 2 3*6 2 10*2 3*6 2 (29)	3*6 2 30*2 3*6 2 (38)	18	0.38143
5	0.2	0.8	0.25 and 0.03 0.97	5*4 10*2 5*4 10*2 5*4 (35)	5*4 10*2 5*4 10*2 5*4 (35)	5*4 30*2 5*4 (40)	27	0.38178
6	0.1	0.9	0.25 and 0.5 0.5	11 9 5*4 11 9 5*4 11 9 (16)	11 9 5*4 11 9 5*4 11 9 (16)	11 9 15*4 11 9 (19)	4	0.37655

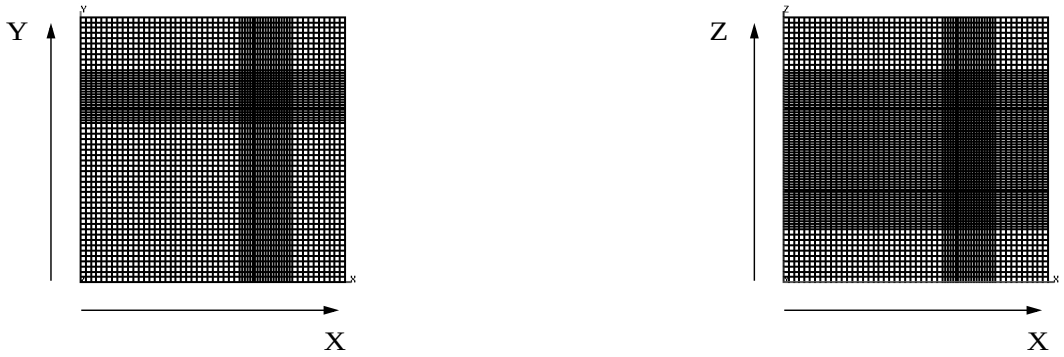

Figure 2. Top and Side Views of Mesh for Problem 2.

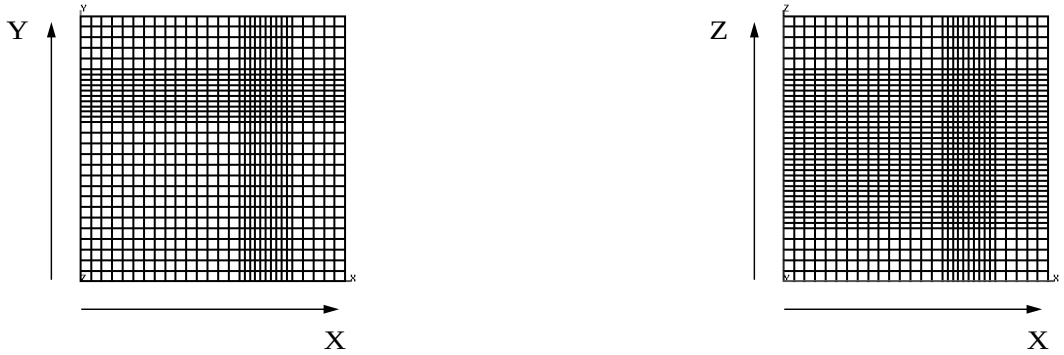

Figure 3. Top and Side Views of Mesh for Problem 3.

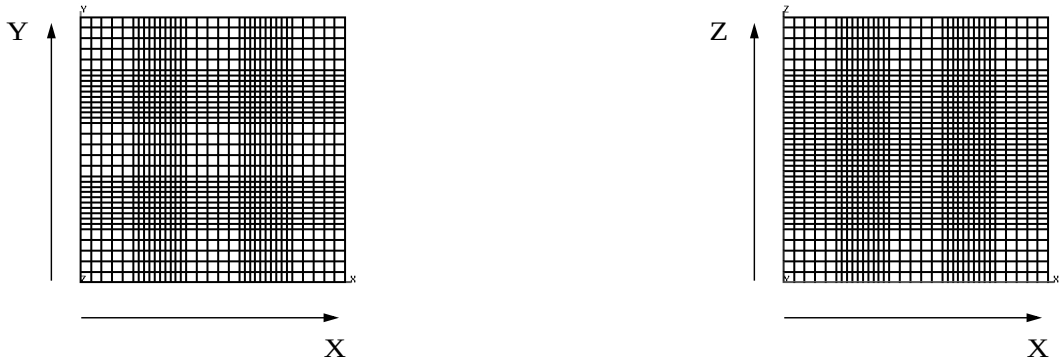

Figure 4. Top and Side Views of Mesh for Problem 5.

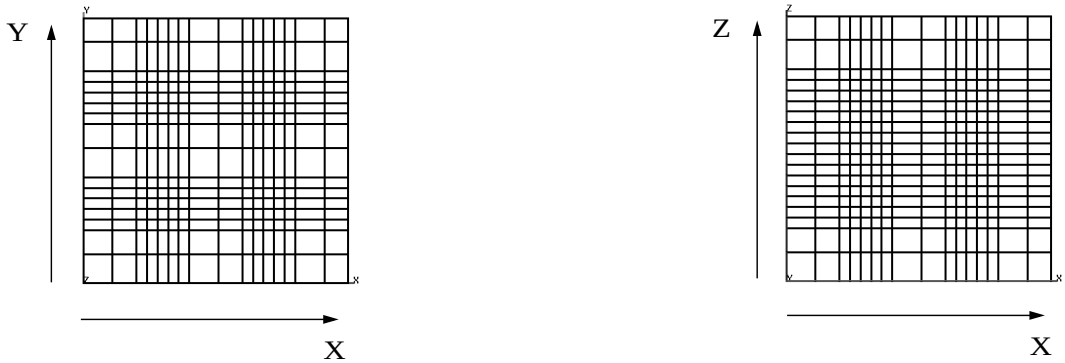

Figure 5. Top and Side Views of Mesh for Problem 6.

In all cases, the cpu time consumed by the mesh coarsening analysis and adjustments took less than one half of a second. Problem 1 is the result of execution with coarsening turned off. Problem 2 is the result of adding a single variable to the input deck to turn coarsening on. In the absence of other user input, the default values in columns 2-4 of the second problem are used by the code. The resulting mesh is described numerically in columns 5-7 of Table 1 and graphically in Figure 2. The coarsened mesh resulted in a value for k-effective that differed by less than 0.03 percent from the original mesh solution but took only 17% as long to run. If 237 seconds is still considered too long, then more coarsening can be achieved by raising the lower bound of the mesh potential against which cells are checked for coarsening. A larger lower bound results in more cells being eligible for combination with neighboring cells. As described in the implementation section, every cell in two neighboring planes has to qualify in order for the planes to be combined. The results for problem 3 indicate that raising the lower bound from the default value of 0.25 to 0.75 results in a mesh with half as many cells in each direction (also depicted in Figure 3). For problem 3, the answer is lower than the original mesh result by less than 0.2 percent and took only 18 seconds to execute.

The first two coarsening attempts emphasized total cross section as the most important factor (90% of the mesh potential weight). If it is determined that the fuel zones should also maintain a relatively fine mesh due to potential gradients in the flux within, then emphasis can be shifted toward the fission cross section as seen in the results for problem 4 (80% weight on fission). The resulting mesh is about the same size as problem 3 but with more cells in the fuel zone and less in the reflector zones. The amount of coarsening in the fuel zones can be reduced further by adjusting the energy weighting to reflect the fact that in this case 97% of the fissions occur in the second group. The results for problem 5 reflect this change in Table 1 and Figure 4. With more emphasis on fission, the mesh is now twice as fine in the fuel and control zones relative to the reflector zones.

If 20 seconds is still too long to wait, then the user can adjust the parameters even further. The input values used for problem 6 result in a coarse mesh (Figure 5) that is solved in only 4 seconds yet gives a k-effective that differs from the original mesh by only about 1.5 percent. This level of accuracy is sufficient to serve the purpose of selecting initial materials and configurations. Hundreds of combinations can be tried in a few hours.

Figuring out an adequate mesh by hand in the above problem would have taken a considerable amount of time, as would designing the input deck for that mesh. Each mesh tried in the cases presented only took a second or two because only one or two parameters had to be changed in the input deck.

4 Conclusion

Preliminary implementation and testing indicates that PARTISN can automatically coarsen an excessively fine mesh while providing reasonably accurate answers. For the problem considered here, “good” automatic neutronics meshes were achieved in far less time than could be achieved by manual input. The advantage should grow with the complexity of the problem. Overall usefulness of the method should grow as new features are implemented.

There are some rather obvious improvements that need to be implemented in order for the method to reach higher levels of usefulness. The ability of the code to refine the original mesh, in addition to the coarsening that is now done, would be a large improvement. Most of the details have been worked out but a few remain, such as redefining processor layouts after a mesh change. The redefinition of these layouts for coarsening only was fairly complicated and will need to be modified for refinement.

Full support of block AMR meshes would also be a significant improvement. Figure 6 depicts the block AMR equivalent of the mesh in Figure 4. This mesh can be obtained now by the user specifying the desired levels for each zone in the original problem. The AMR solver implemented by Baker (Baker, 1998) in PARTISN, requires uniform meshes within each AMR block for efficiency of the parallel sweep. Each block contains 2^L cells in each direction, where L is the level of the block. A

level 0 block contains a single cell where as a level 3 block contains a matrix of 8 cells in each direction. As long as the specified levels are equal to or finer than the original mesh, the mesh coarsening routines will automatically adjust the original mesh to conform to the user-specified levels. Much more useful would be a routine which utilizes the mesh potential to automatically determine how a mesh is broken up into blocks and the appropriate level for each block.

Figure 6. Top and Side Views of Block AMR Mesh.

Energy dependent meshes are another potential improvement. The ability to specify a unique mesh for each energy group may result in a far more efficient calculation because cross sections vary greatly with energy.

Control over the final mesh's aspect ratio is also important for efficiency and accuracy. Cells which resemble long thin rods are problematic for diamond differencing, and other simple schemes. They tend to overestimate leakage through the long dimension when fluxes are positive and underestimate leakage when set to zero fixup is used. Enforcing a limit on aspect ratios will reduce the amount of coarsening that can be accomplished, which is another reason why AMR meshes are useful.

A Appendix

PARTISN input deck used to generate the results presented in Section 3:

```
    0
 igeom=14 ngroup=2 isn=8 niso=3 mt=3 nzone=3 im=5 it=100 jm=5 jt=100
 km=3 kt=100 maxscm=1100000
    t
 xmesh=0.0 5.0 10.0 15.0 20.0 25.0  xints=20 20 20 20 20
 ymesh=0.0 5.0 10.0 15.0 20.0 25.0  yints=20 20 20 20 20
 zmesh=0.0 5.0 20.0 25.0  zints=20 60 20
 zones=5r3; 4y1; 5r3; 3 1 3r3; 5r3; 3r3 2 3; 5r3; 5r3; 4y1;
    t
 lib=odninp ihm=6 iht=3 ihs=4 ititl=1 t
/  absorption     fission          total          g->g scatt      g->g+1
core
 0.00852709     0.01818638   0.223775     0.18332981     0.0              0.0
 0.15819600     0.58036600   1.038640     0.59025600     0.0228253        0.0
contrl
 0.3488780       0.0             1.7046500    1.3544820      0.0              0.0
 3.6444800       0.0             4.3492000    0.7047160      1.2909220e-3     0.0
refl
 0.000416392     0.0             0.250367     0.193446       0.0              0.0
 0.020299900     0.0             1.644820     1.624520       0.0565042        0.0

 matls=mat1 iso1 1.0; mat2 iso2 1.0; mat3 iso3 1.0;
 assign=zone1 mat1 1.0;
          zone2 mat2 1.0;
          zone3 mat3 1.0; t
```

```
 ievt=1 ibl=1 ibb=1 ibfrnt=1 norm=1 iitl=1 oitm=25
 epsi=5.0e-5 chi=1.0 0.0
 srcacc=dsa diffsol=cg1l
 mshcol=2
 mshcasg=0.9
 mshcbnf=0.1
 mshcrbn=0.75
 mshcewt=0.03 0.97
 extras=2r0 1 f0
    t
```

References

[Alcouffe et al., 1998] Alcouffe, R. E., Baker, R. S., Turner, S. A., Dahl, J. A., PARTISN, LA-CC 98-62, Los Alamos National Laboratory, Los Alamos NM, 87545, 1998.

[Baker, 1998] Baker, R. S., Software Requirements Specification for Block AMR in PARTISN, Memorandum XTM:98-78(U), Los Alamos National Laboratory, Los Alamos NM, 87545, October 15, 1998.

[Takeda, 1991] Takeda, T., Ikeda, H., 3-D Neutron Transport Benchmarks, OECD/NEA Committee on Reactor Physics, NEACRP-L-330, March 1991.